Extraction of Key Words from News Stories

Ramesh Nallapati, James Allan, Sridhar Mahadevan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{nmramesh, allan,mahadeva}@cs.umass.edu

ABSTRACT

In this work, we consider the task of extracting key-words such as key-players, key-locations, key-nouns and key-verbs from news stories. We cast this problem as a classification problem wherein we assign appropriate labels to each word in a news story. We considered statistical models such as naïve Bayes model, hidden Markov model and maximum entropy model in our work. We have also experimented with various features. Our results indicate that a maximum entropy model that ignores contextual features and considers only word-based features combined with stopping and stemming yields the best performance. We found that extraction of key-verbs and key-nouns is a much harder problem than extracting key-players and key-locations.

1. INTRODUCTION

Key-word extraction in news stories is the process of identifying important words in the story that bear most of the topical content of a news story. For example, in a news story that reports the *9-11* attacks, key words could be 'twin-towers', 'collapse', 'hijack', 'jet', 'terrorists', 'attack', etc. Clearly, these words capture the topical information and convey us the essence of the news story.

The task of key-word extraction finds applications in several other IR-related tasks such as Topic Detection and Tracking (TDT) [1], summarization and ad-hoc retrieval. TDT concerns itself with organizing news stories by the events that they discuss. Our preliminary experiments show that accurate extraction of key-words from news stories can aid in better organization of news stories by their events. Summarization, on the other hand deals itself with automatically generating human-readable short summaries of documents. We believe identifying key-words in news stories is the first step in building an effective summary of a document. In ad-hoc retrieval, indexing a document by its key-words and not by the whole bag of words may help make the search faster and more precise.

The rest of the document is organized as follows. In section 2, we present the past work in this and related tasks. In section 3, we describe the detailed description of the task and the corpus and the evaluation criteria. In section 4, we describe the models we used and present results obtained. Section 5 concludes our work with a few notes on future work.

2. RELATED WORK

There is surprisingly very little work on key-word extraction available in the literature. In a paper closest to the current work, Turney [12] extracts key phrases from technical papers using a decision tree based on features like word-length, parts-of-speech, occurrence statistics, etc. Turney also presented an improved algorithm that uses a rule-based extractor that learns its parameters in a supervised fashion on a training set using a genetic algorithm.

In other related work, Krulwich and Burkey [4] use heuristics to extract significant phrases from a document. The heuristics are based on syntactic clues, such as the use of italics, the presence of phrases in section headers, and the use of acronyms. Steier and Belew [11] use the mutual information statistic to discover two-word key-phrases. However, both the algorithms tended to produce low precision performance.

Another body of related work is the Message Understanding Conference (1991-95) [15] sponsored by ARPA wherein information extraction systems are evaluated with corpora in various topic areas, including terrorist attacks and corporate mergers. An MUC extraction system seeks specific information in a document, according to predefined guidelines. The guidelines are specific to a given topic area. For example, if the topic area is news reports of terrorist attacks, the guidelines might specify that the information extraction system should identify (i) the terrorist organization involved in the attack, (ii) the victims of the attack, (iii) the type of attack (kidnapping, murder, etc.), and other information of this type that can be expected in a typical document in the topic area. Most MUC systems are manually built for a single topic area, which requires a large amount of expert labour. The highest performance at the Fifth Message Understanding Conference (MUC-5, 1993) was achieved at the cost of two years of intense programming effort. However, recent work by Soderland has demonstrated that a learning algorithm can perform as well as a manually constructed system [10]. They use decision tree induction as the learning component in their information extraction system.

In our task of identifying key words from news stories, we are interested in identifying four classes of key-words namely key-players, key-locations, key-verbs and key-nouns. We believe our task lies somewhere in the middle in a spectrum that ranges from the binary classification of words into key and non-key classes as done in [12, 4, 11] and the template filling task of MUC [15]. However, our task is more general than MUC's task in the sense that the news stories we use are not restricted to a specific genre.

CIIR technical Report # IR-345

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **2004**	2. REPORT TYPE	3. DATES COVERED **00-00-2004 to 00-00-2004**

4. TITLE AND SUBTITLE
Extraction of Key Words from News Stories

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Massachusetts Amherst,Department of Computer Science,140 Governors Drive,Amherst,MA,01003

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In this work, we consider the task of extracting key-words such as key-players, key-locations, key-nouns and key-verbs from news stories. We cast this problem as a classification problem wherein we assign appropriate labels to each word in a news story. We considered statistical models such as naive Bayes model, hidden Markov model and maximum entropy model in our work. We have also experimented with various features. Our results indicate that a maximum entropy model that ignores contextual features and considers only word-based features combined with stopping and stemming yields the best performance. We found that extraction of keyverbs and key-nouns is a much harder problem than extracting keyplayers and key-locations.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**6**	

3. TASK DESCRIPTION

In this section, we lay out the framework of the task, describe the corpus and define evaluation criteria. We start with defining what we exactly mean by key-words.

3.1 Key-words and their classes

The goal of the present task is to extract key words in a news story that bear most of the information content of the topic of the news story. In defining what words constitute the key words, we bank on the paradigm of TDT [1] which defines a topic as a "seminal activity or event, along with all directly related events and activities." Furthermore, an event is defined as "something that happens at a specific time and place along with all necessary preconditions and unavoidable consequences." We concluded from the above definition that words that answer the questions 'who?', 'where?', 'what?' and 'when?' are the key-words in a news story since they help us define the event of the story and hopefully, the topic. Of these questions, we ignore the 'when?' question, since we believe it is easy enough to extract from an off-the-shelf named-entity tagger. Accordingly, we define the following classes of words in a news story:

1. **Key-player**: This class represents a person or an organization or a group that is central to the story.For example, in a story about Daniel Pearl's Kidnap and murder, Daniel Pearl, America and Pakistan and Kidnapers would be the key-players. In a story about an earthquake relief operations, Red Cross could be the key-player. Clearly, this requires us to ignore occurrences of entities such as witnesses, spokespersons and reporters' names from this class.

2. **Key-location**: Any location occurring in the story that is connected to the event is a key-location. For example, occurrences of 'U.S.', 'New York' and 'WTC' are all event locations in a story that discusses the *911-attack*. Other occurrences of location such as the reporting location are not to be classified under this class.

3. **Key-verb**: A verb occurring in the story that best describes an action occurring in the story is is a *Key verb*. For example, in a story about the 'war on terror' some of the key verbs could be *bombed*, *attacked*, *killed*, etc.

4. **key-noun**: A noun occurring in the news story that best describes the event in question belongs to the *Key noun* class. For example, in a story that details the *911 attack*, nouns such as *collapse*, *destruction*, *attack* are key-nouns. In a story about earthquake, the noun *earthquake* itself could be a key noun. In Daniel Pearl's story, nouns such as *kidnap* and *murder* could be the key nouns.

5. **None**: This is the class of words that do not belong to any of the categories mentioned above.

Words in the classes *key-verb* and *key-noun* are expected to answer the 'what?' question while the classes *key-player* and *key-location* answer the questions 'who?' and 'where?' respectively.

3.2 Corpus

We hired three undergraduate students, one of them a student of Journalism and the other two, of Computer Science to annotate a subset of the TDT2 corpus with key-words. We have used *Alembic Work Bench* [13] as an annotation interface tool. The students were asked to read each story completely and understand thoroughly the contents of the story before tagging the key-words. The annotators were asked to tag all occurrences of a key-word in a news story with its class. Also, they were restricted from tagging a single phrase or word by more than one class. They were however allowed to tag each story with zero or more instances of each class. After the tagging by annotators, we did some cleaning up to make sure there were no mistakes. For example, we did the following corrections:

- If the annotator-tags crossed with the automatic named-entity tags generated by Identifinder [3], we re-aligned them.
- If word is tagged as a key-player or a key-location and it is not a noun, we removed the tags.
- We made sure that the part-of-speech of the words tagged as key-nouns and key-verbs are nouns and verbs respectively.

We created an annotated corpus that comprises 974 stories from 59 topics. We split them into 593 training stories from 32 topics and 381 test stories from the remaining 27 topics. Note that there is no overlap between training and testing topics. This ensures that our learning algorithm is general enough to handle varied topics.

3.3 Evaluation

We define the task of extracting the key-words as one of assigning each word in a news story a label from the set defined by the four classes. Although some of the annotator-tags are assigned to phrases such as 'New York', 'World Trade Center' or 'United States' etc., we nevertheless treat words as our smallest units of labeling for the sake of simplicity. For instance, if a *key-location* tag is assigned to the phrase 'United States of America', we assume each of the words in the phrase has the tag *key-location*. This may not be the best approach to adopt, but we believe it is a strategy that makes the task simple and serves as a good starting point. We also believe that such examples are not very frequent and may not significantly alter the results of our experiments.

We use a supervised learning algorithm that learns its parameters from the training set and assigns the best labels to the words in the test set. We measure the performance of the algorithm in relation to the tags assigned by the annotators. For each class i, we measure precision (P_i) and recall (R_i) defined as follows:

$$\begin{aligned} Prec_i &= \frac{\#(assigned_i \ and \ correct_i)}{\#(assigned_i)} \\ recall_i &= \frac{\#(assigned_i \ and \ correct_i)}{\#(correct_i)} \end{aligned} \tag{1}$$

where $\#(assigned_i)$ is the number of words assigned class i by the algorithm and $\#(correct_i)$ is the number of words that are actually in class i as per the annotations. We further compute the averages of precision (P_{avg}) and recall (R_{avg}) over all classes (excluding the *None* class) and finally compute a single evaluation measure called the F1-measure which is the harmonic mean of average precision and average recall as shown below:

$$F1 = \frac{2P_{avg}R_{avg}}{P_{avg} + R_{avg}} \tag{2}$$

4. EXPERIMENTS AND RESULTS

In this section, we present the supervised learning models we considered and the results we obtained on each one of them.

4.1 Naïve Bayes' Classifier

In this model, we treat each word as an I.I.D. sample and classify each word separately. We considered the word, its part-of-speech as determined by jtagger *jtag*, its named-entity tag as determined by *BBN*'s name-finder [3], the last three nodes in the path from the root to the word in the parse tree of the corresponding sentence as determined by the *Applie Pie Parser* [9] as the features in the model. A few examples of words, their features and the annotators' tags from a single sentence in a news story are shown in figure 1.

word	POS	NE	Parse	Label
president	np	None	s-npl-nnpx	None
fidel	np	person	s-npl-nnpx	Key-player
ramos	np	person	s-npl-nnpx	Key-player
has	hvz	None	s-s-vp	None
urged	vbn	None	s-vp-vp	Key-verb
king	np	None	ss-npl-nnpx	None
norodom	np	person	ss-npl-nnpx	Key-player
sihanouk	np	person	ss-npl-nnpx	Key-player
to	to	None	vp-ss-vp	None
return	vb	None	ss-vp-vp	Key-verb
to	toin	None	vp-vp-pp	None
cambodia	np	location	vp-pp-npl	Key-location

Figure 1: words, their features and their key-word tag extracted in a sentence

The classes are the four *key* labels and additional class called 'NONE' which represents absence of any label. A naïve Bayes' classifier considers the features to be conditionally independent of each other given the class and can be represented graphically as shown in figure 2.

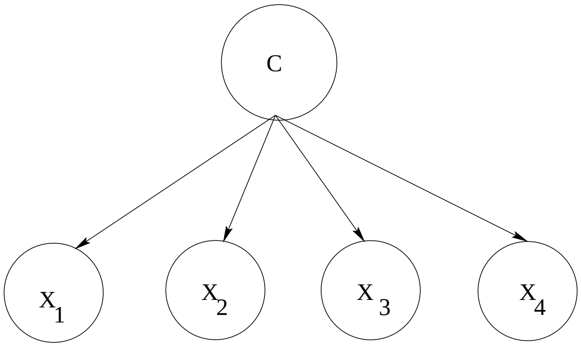

Figure 2: Graphical representation of the Naïve Bayes' classifier

The discriminant function of each class is given by the log of the posterior probability of the class as shown by the following equation:

$$g(c) = logP(c|x_1, .., x_n) = (\sum_{i=1}^{n} logP(x_i|c) + logP(c)) + K \tag{3}$$

where n is the number of features and

$$K = -\sum_c \sum_i logP(x_i|c)P(c)$$

is a normalizing constant. The prior probabilities $P(c)$ and the class conditionals $P(x_i|c)$ are computed from the smoothed maximum likelihood estimates from the training set as shown below:

$$P(x_i|c) = \lambda \frac{n_t(x_i, c)}{n_t(c)} + (1 - \lambda) \frac{\sum_c n_t(x_i, c)}{\sum_c n_t(c)} \tag{4}$$

where $n_t()$ is the number of times the argument occurs in the training corpus. We smooth the class conditional frequencies with frequencies over the entire training set. This helps reduce over fitting and improve generalization. We set $\lambda = 0.9$. The class priors $P(c)$ is simply given by the relative frequency of its occurrence in the training set as shown below:

$$P(c) = \frac{n_t(c)}{\sum_c n_t(c)} \tag{5}$$

The results are reported in the form of a confusion matrix in figure 3. Each row tells us how a given label is classified by the classifier. For instance, the first row tells us that of all the words that belong to the class *None*, 75233 are classified as *None*, 3220 as *Key-noun*, 4575 as *Key-verb* , *etc.* Note that we ignore the class *None* in computing the average precision and recall values.

Ref↓Hyp→	**None**	**K.Noun**	**K.Verb**	**K.Loc**	**K.Plyr**
None	75233	3220	4575	1102	2367
K.Noun	1182	665	2	83	118
K.Verb	199	0	296	0	0
K.Loc	2	0	0	781	78
K.Plyr	4004	127	182	401	4172
Rel.	86497	2050	495	861	8886
Ret'ved	80620	4012	5055	2367	6735
Prec.	-	0.16	0.05	0.32	0.61
Recall	-	0.32	0.59	0.90	0.46

Avg Pr	**Avg Rec**	**F1**			
0.29	0.57	0.38			

Figure 3: Results of the naïve Bayes' Classifier

From the table, it is clear that the classifier is very imprecise although the recall is reasonable. In particular, the classes key-verbs and key-nouns seem very hard to classify. We try to incorporate additional contextual information into the classifier by constructing a conditional naïve Bayes' classifier as described in the following subsection.

4.2 Conditional Naïve Bayes' classifier

In this model, we still consider each word and its features to be I.I.D., but we condition the class of each sample c on the class of the previous sample(word) c_{-1}. Note that this forces us to classify each word in the order of its occurrence because we use the best label of the previous word as the value of the conditioning variable in the current classification. The graphical representation of the conditional naïve Bayes' classifier is shown in figure 4. The discriminant function is as shown below:

$$\begin{aligned} g(c) &= logP(c|x_1, ..., x_n, c_{-1}) && (6) \\ &= \sum_i logP(x_i|c, c_{-1}) + logP(c|c_{-1}) + K && (7) \\ &= \sum_i logP(x_i|c) + logP(c|c_{-1}) + K && (8) \\ & && (9) \end{aligned}$$

where step 7 comes from applying Bayes' rule and step 8 follows from conditional independence of the feature variables x_i from the previous class label c_{-1} given the current class label c (see figure 4). The estimates of class conditionals are same as in equation 4, but the prior probabilities of the class $p(c)$ are conditioned on the previous class c_{-1}, hence it is estimated as follows:

$$P(c|c_{-1}) = \frac{n_t(c_{-1}, c)}{n_t(c_{-1})} \tag{10}$$

Here $n_t(c_{-1}, c)$ is the number of adjacent examples in the training set that have the labels c_{-1} and c in that order respectively. The results from this classifier are shown in table 5.

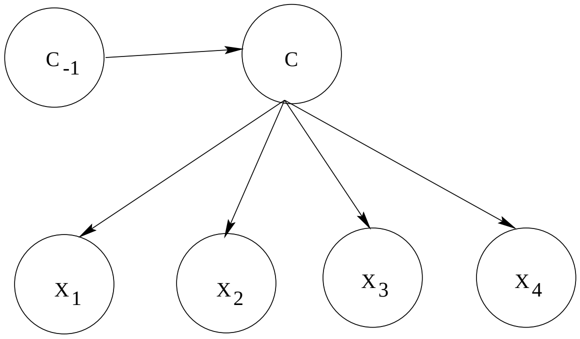

Figure 4: Graphical representation of the Conditional Naïve Bayes' classifier

Ref↓Hyp→	None	K.Noun	K.Verb	K.Loc	K.Plyr
None	75069	3214	4624	1152	2438
K.Noun	1189	660	2	87	112
K.Verb	201	0	294	0	0
K.Loc	6	0	0	772	83
K.Plyr	4037	127	184	424	4114
Rel.	86497	2050	495	861	8886
Ret'ved	80502	4001	5104	2435	6747
Prec.	-	0.16	0.05	0.31	0.60
Recall	-	0.32	0.59	0.89	0.46
Avg Pr	**Avg Rec**	**F1**			
0.28	0.56	0.38			

Figure 5: Results of the conditional naïve Bayes' classifier

We see that there is no significant change in performance as compared to the naïve Bayes' model. We could explain this in two different ways: either the information about the previous class is inconsequential to the current classification or that imperfect classification of the previous word hurts the classification of the current word. To understand the actual reasons, we implemented a hidden Markov model wherein we compute the best sequence of classification for the whole sequence of words in a sentence. The hypothesis is that if the hidden Markov model improves on the performance, it could mean that contextual information is important and the reason for the failure of the conditional model is because of imperfect contextual information.

4.3 Hidden Markov Model

In this model, we consider not words, but sentences to be I.I.D. samples. Hence the problem now is to estimate the best sequence of class-labels corresponding to the sequence of features as shown below:

$$\text{Most likely label sequence } \mathbf{C} = \arg\max_{\mathbf{C}} \; P(\mathbf{C}|\mathbf{X}) \tag{11}$$

where $\mathbf{C}$ is the sequence of class-labels corresponding to the sequence of word-feature vectors of the sentence represented by $\mathbf{X}$. We use Bayesian inversion to obtain the following:

$$\arg\max_{\mathbf{C}} \; P(\mathbf{C}|\mathbf{X}) = \arg\max_{\mathbf{C}} \; P(\mathbf{X}|\mathbf{C})P(\mathbf{C}) \tag{12}$$

We now assume that the feature vector corresponding to the i-th word in the sentence, $\mathbf{X_i}$, depends only on the class-label generating it and the class-label c_i in turn depends only on the previous label c_{i-1}. This is graphically represented in figure 6. Following the above assumptions, the posterior $P(\mathbf{C}|\mathbf{X})$ can be approximated as:

$$\arg\max_{\mathbf{C}} \; P(\mathbf{C}|\mathbf{X}) = \arg\max_{\mathbf{C}} \; \prod_{i=1}^{n} P(c_i|c_{i-1}) \prod_{j=1}^{m} P(x_{ij}|c_i) \tag{13}$$

where n is the sentence length and m is the number of features of each word, in our case it is 4.

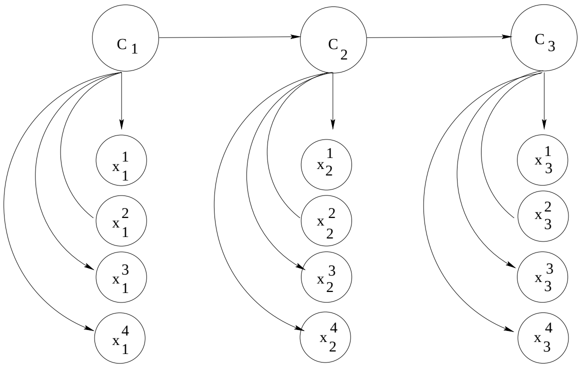

Figure 6: graphical representation of the HMM

The class-conditionals and the class-priors are estimated in a similar fashion as described in the conditional Naïve Bayes” model described in sub-section 4.2. Once we have the probabilities, it is trivial to compute the best sequence using the standard Viterbi algorithm [8].

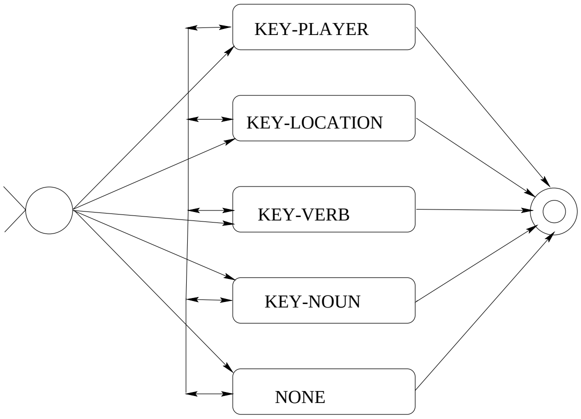

Figure 7: The State-Transition diagram of the HMM

In figure 7 we have shown the state diagram of the HMM. Here the states are our class-labels shown by the squares, the start state is indicated by a circle and an incoming arrow and the end state is represented by concentric circles. The starting state represents the beginning of the sentence and the end state represents the end of the sentence. The HMM transits between the states any number of times, producing an observation each time, which is the feature vector corresponding to a given word, until it reaches the end state.

The HMM is fully connected, hence it is free to transit to any state from any given state.

Ref↓Hyp→	None	K.Noun	K.Verb	K.Loc	K.Plyr
None	83529	431	285	622	1630
K.Noun	1936	74	0	10	30
K.Verb	443	0	52	0	0
K.Loc	372	0	0	436	53
K.Plyr	6478	21	7	187	2193
Rel.	86497	2050	495	861	8886
Ret'ved	92758	526	344	1255	3906
Prec.	-	0.14	0.15	0.34	0.56
Recall	-	0.03	0.10	0.50	0.24
Avg Pr	**Avg Rec**	**F1**			
0.30	0.22	0.25			

Figure 8: Results of the HMM classification

Figure 8 presents the results from the HMM classification. Disappointingly, the HMM performs worse than the naiv̈e Bayes' or the conditional naïve Bayes' classifiers. This shows us that keywords do not really form a sequence data. Hence we believe this is an evidence that context may not play an important role in the classification of key-words. We surmise that other features of term statistics like frequency of occurrence in the news story and general English frequency, may be important in key-word extraction. Hence we turn our attention to such features in our next attempt. We choose the the maximum entropy model for the new experiments considering its capability of modeling arbitrary features making the least modeling assumptions. Following the experience from the previous models, we focus on just the features of the word and ignore its context in defining the model's features.

4.4 Maximum entropy model

In maximum entropy, we use the training data to set constraints on the conditional distribution. Each constraint expresses a characteristic of the training data that should also be present in the learned distribution [6]. We let any real valued function of the word and the class be a feature, $f_i(w;c)$. Maximum entropy allows us to restrict the model distribution to have the same expected value for this feature as seen in the training data. Thus, we stipulate that the learned conditional distribution $P(c|w)$ must have the property:

$$\sum_{w,c} \hat{p}(w,c) f_i(w,c) = \sum_{w,c} \hat{p}(w) p(c|w) f_i(w,c) \qquad (14)$$

When constraints are estimated in this fashion, it is guar anteed that a unique distribution exists that has maxi mum entropy. Moreover, it can be shown [2] that the distribution is always of the expo nential form:

$$P(c|w) = \frac{1}{Z(w)} exp(\sum_i \lambda_i f_i(w,c)) \qquad (15)$$

where $Z(w)$ is a normalizing constant given by:

$$Z(w) = \sum_c exp(\sum_i \lambda_i f_i(w,c)) \qquad (16)$$

When the constraints are estimated from labeled training data, the solution to the maximum entropy problem is also the solution to a dual maximum likeli hood problem for models of the same exponential form. Additionally, it is guaranteed that the likelihood surface is convex, having a single global maximum and no lo cal maxima. Thus any hill climbing algorithm performed on an initial guess of an exponential distribution of the correct form is guaranteed to converge to the maximum likelihood solution for exponential models, which will also be the global maximum entropy solution.

We used *Mallet* [14] to implement the maximum entropy model. Apart from the four features presented in figure 1, we used the following additional set of features in the classifier:

1. *Term frequency ratio*: This feature tells us the relative frequency of the word with respect to the most frequent word in the news story. We believe this could be an important feature in determining the key-words in a news story.

2. *Inverse-document frequency (idf)*: This is the defined as the negative logarithm of the proportion of documents a word occurs in the collection. We compute idf weights for words in test and training sets separately with respect to their own collections. Commonly used in IR, higher the idf-weight, more is our confidence that the word is importance to the story, because it occurs less frequently in the collection.

3. *Position in the story*: We divided the story into four quarters based on the document length and the value of this feature tells us the quarter in which the word occurs. We noticed that typically key-words occur more frequently in the beginning of the story than otherwise, hence we believe this feature may help us distinguish key-words from others to some extent.

Figure 10 presents the results from the maximum entropy classifier. The maximum entropy succeeds in improving the overall precision but at the expense of a considerable loss in recall as compared to the conditional naiv̈e Bayes' classifier. Hence the single point F1-measure ends being almost the same.

Ref↓Hyp→	None	K.noun	K.verb	K.loc	K.Plyr
None	83614	596	374	582	1331
K.noun	1738	262	0	8	42
K.verb	407	0	87	0	1
K.location	290	0	0	512	59
K.Plyr	4361	8	1	211	4305
Rel.	86497	2050	495	861	8886
Ret'ved.	90410	866	462	1313	5738
Prec.	-	0.30	0.18	0.38	0.75
Recall	-	0.12	0.17	0.59	0.48
Avg Pr	**Avg Rec**	**F1**			
0.40	0.34	0.37			

Figure 9: Results of the maximum entropy classifier

We think that one of the reasons for the lack of improvement in performance is the ineffective estimation of the features such as the term frequency ratio. For example, in most news stories, the words *the*, *an* and *of* end up having the highest term frequency ratio. Hence we decided to remove stop words since we are ignoring the context in any case. We also stemmed the words to their root forms using Porter stemmer [7] so that similar words are collapsed together and may help in improving the parameter estimates. We also collapsed proper nouns that span multiple words into single entities. For example *New York* and *United States of America* are treated as single words each instead of a sequence of words.

The results of the maximum entropy classification on stopped and stemmed data are shown in figure 10.

It is clear that stemming and stopping combined with proper name collapsing has helped improve both recall and precision. The

Ref↓Hyp→	**None**	**K.noun**	**K.verb**	**K.Loc**	**K.Plyr**
None	36703	677	424	296	878
K.noun	1522	325		30	79
K.verb	326		148		
K.Loc	125			470	22
K.Plyr	241			236	2434
Rel.	38978	1956	474	617	2911
Ret'ved.	38917	1002	572	1032	3413
Prec.	-	0.32	0.25	0.45	0.71
Recall	-	0.16	0.31	0.76	0.83

Avg Pr	**Avg Rec**	**F1**			
0.43	0.51	0.47			

Figure 10: Results of the maximum entropy classifier on stopped and stemmed data

Label type	**Train Set**	**Test Set**
K.Plyr/{Pers. or Org.}	4057/5484=74%	2911/3752 = 78%
K.Loc/Loc.	1480/2605=57%	617/1944 = 32%
K.Noun/Noun	5605/44783 = 12%	1956/26736 = 7%
K.Verb/Verb	3030/15716 = 9%	474/8882 = 5%

Figure 11: Proportion of key-labels in words that belong to their respective linguistic classes

overall performance is still below 50% but the performance on *key-players* and *key-locations* seems satisfactory. The other two classes, namely, *key-nouns* and *key-verbs* have proved very hard to classify. The table in figure 11 illustrates the reason behind this performance discrepency between the labels. Each row in the table shows the statistics of each key label as percent of words that are in the same linguistic class. For example we computed the ratio of total number of *Key Players* to the total number of *persons* and *organizations* in the test and train sets. The table clearly shows that there are far fewer proporption of *key-nouns* and *key-verbs* than the other two classes, which we believe makes it harder to recognize them.

5. CONCLUSIONS AND FUTURE WORK

In this work, we have built statistical models to extract key-phrases from news stories. Our work is different from the binary classification of words into key and non-key words in that we label the key-words with their specific types too. Hence our task is a multi-class classification problem. It is also unlike the MUC-extraction task in the sense that we do not restrict the domain to any specific genre. After experimenting with a conditional naïve Bayes' classifier and hidden Markov model, we found that context of words does not play an important role in determining key-words. Using an enhanced feature set that comprises the term frequency, inverse document frequency and position of the word combined with stopping and stemming has yielded the best performance.

There is a lot that remains to be done as part of the future work. In particular, we believe that better preprocessing can further improve the performance of the model. For example normalizing different representations of the same entity (such as United States of America, United States, US, USA, etc.) into a single form can aid the performance. We are also considering making use of an external knowledge-base such as the web in identifying the key words in news stories. Also, it is clear from the results that classification of key-nouns and key-verbs is a harder problem than key-players and key-locations. Hence it makes sense to classify the key-players and key-locations first and then use this information to classify key-nouns and key-verbs in the second stage. This conjecture is based on the premise that key-nouns and key-verbs that typically express actions are usually linked to the key-locations and key-players. Hence classification of the latter two may aid in classifying the former. We intend to pursue this path as part of our future work.

Acknowledgments

We would like to thank Andrew McCallum and Victor Lavrenko for their valuable comments. This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

6. REFERENCES

[1] Allan, J. *Topic Detection and Tracking - Event-based Information Organization*, Kluwer Academic Publishers, 2002.

[2] Berger, Adam L., Della Pietra, Stephen A. and Della Pietra, Vincent J., A Maximum Entropy Approach to Natural Language Processing, *Computational Linguistics*, 1996, vol. 22(1), p39-71.

[3] Bikel, D. M., Miller, S., *et al*, Nymble: a high-performance learning name-finder, *Proceedings of ANLP-97*, p 194-201, 1997.

[4] Krulwich, B., and Burkey, C., Learning user information interests through the extraction of semantically significant phrases, In *M. Hearst and H. Hirsh, editors, AAAI Spring Symposium on Machine Learning in Information Access*, 1996.

[5] Manning, C. D. and Schutze, H., *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

[6] Nigam, K., Lafferty, J. and McCallum, A., Using maximum entropy for text classification, *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61-67, 1999.

[7] Porter, M. F. An algorithm for suffix stripping, *Program*, 14(3):130-137, 1980.

[8] Rabiner, L. R., A tutorial on hidden Markov models, *Proceedings of the IEEE*, vol. 77, pp. 257-286, 1989.

[9] Sekine, S. and Grishman, R., A Corpus-based Probabilistic Grammar with Only Two Non-terminals, *In Proceedings Fourth IWPT, Prague, Czech Republic*, 1995.

[10] Soderland, S. and Lehnert, W., Wrap-Up: A trainable discourse module for information extraction, *Journal of Artificial Intelligence Research*, 2, 131-158.

[11] Steier, A. and Belew, R. K., Exporting phrases: A statistical analysis of topical language. Document Analysis and Information Retrieval (DAIR) Conference, 1993.

[12] Turney, P., Learning to extract key phrases from text, *Technical Report ERB-1057, National Research Council, Institute for Information Technology*, 1999.

[13] Alembic Work Bench, *http://www.mitre.org/technology/alembic-workbench/*

[14] McCallum, A. K., MALLET: A Machine Learning for Language Toolkit. *http://www.cs.umass.edu/ mccallum/mallet*, 2002.

[15] MUC-6, *Proceedings of the Sixth Message Understanding Conference*, California: Morgan Kaufmann, 1995.